高帧频面探测器分布式数据获取软件研究

商琨琳[1]　李正恒[1]　鞠旭东[1]　周悦[1]　怀平[1,2]　刘志[1]
1（上海科技大学　上海 201210）
2（中国科学院上海高等研究院　上海　201210）

摘要　高帧频面探测器是上海硬 X 射线自由电子激光装置（Shanghai HIgh repetitioN rate xfel and Extreme light facility，SHINE）上开展相干衍射成像、串行晶体学等科学实验的关键设备。面探测器所产生的海量数据的高速传输和处理对数据获取（DAQ）软件提出了极大挑战。为满足百万像素规模、10kHz 帧频的面探测器数据传输要求，DAQ 需实现≥20 GB·s^{-1} 的稳定数据传输能力。本研究开发了基于 C++的分布式 DAQ 软件，旨在实现高速并行的数据读出，满足高帧频面探测器的数据获取。本研究成功实现了 4 个节点 20 GB·s^{-1} 的高通量数据传输和事例重建能力，并对数据刻度、无损压缩等数据预处理功能以及软件整体的分布式运行进行了实现和测试。本研究可以为采用面探测器的相关实验的超高通量数据获取提供必要支持。

关键词　面探测器；数据获取；数据压缩；高帧频；分布式计算
中图分类号　TP319

Research on distributed data acquisition software for high frame rate area detectors

SHANG Kunlin[1]　LI Zhengheng[1]　JU Xudong[1]　ZHOU Yue[1]　HUAI Ping[1,2]　LIU Zhi[1]
1(*ShanghaiTech University, Shanghai 201210, China*)
2(*Shanghai Advanced Research Institute, Chinese Academy of Sciences, Shanghai 201210, China*)

然科学基金委国家重大科研仪器研制项目——超高帧频大动态范围 X 射线探测系统(22127901)资助
第一作者：商琨琳，男，1997 年出生，2020 年毕业于东北大学秦皇岛分校，现为硕士研究生，研究领域为探测器数据采集
通讯作者：李正恒，E-mail: lizhengheng@shanghaitech.edu.cn
收稿日期：20XX-00-00，修回日期：20XX-00-00

Abstract [Background]: High frame rate area detectors play a crucial role in experiments such as coherent diffraction imaging and serial crystallography conducted at the Shanghai HIgh repetitioN rate xfel and Extreme light facility (SHINE). Their high data transmission rates and extensive processing needs pose grand challenges for the data acquisition system. **[Purpose]:** To support the continuous operation of a megapixel detector working at 10kHz frame rate, the DAQ software must deliver a data throughput of no less than 20 GB·s^{-1}. **[Methods]:** In order to meet the requirement of high throughput data readout and processing of many detector modules, this research developed a distributed DAQ software based on C++ language, which is designed to run on multiple servers in parallel. This research adopts a memory based MapReduce-like method to realize the online event reconstruction. And also tested the Bitshuffle+LZ4/ZSTD data compression algorithm. **[Results]:** The software successfully achieved a throughput of 20GB·s^{-1} for data transmission and event reconstruction by using 4 DAQ servers. It also includes the implementation and testing of various functions, such as data calibration, lossless compression, and the distributed operation capability of the whole system. **[Conclusions]:** This research will provide essential support for some related experiments using area detectors which require high throughput data acquisition

Key words Area detector; Data acquisition; Data compression; High repetition rate; Distributed computing

X 射线自由电子激光（X-ray Free Electronic Laser，XFEL）是当前最新一代的先进光源大科学平台装置，具有极为优异的光子性能。以建设中的上海硬 X 射线自由电子激光装置（Shanghai HIgh repetitioN rate xfel and Extreme light facility，SHINE）为例，其具有超高帧频（最高 1MHz）、超高亮度（≥10^{12} 光子/脉冲）、超短脉冲（约百飞秒）、超强相干等显著特点。国际上，美国、德国、日本、瑞士等众多科技强国均在开展 XFEL 的研究和建设。XFEL 在物理、化学、材料、生命等科学领域应用非常广泛[1-2]。例如基于 XFEL 的飞秒串行晶体学（Serial Femtosecond Crystallography, SFX）实验技术，无需冷冻在室温下即可解析蛋白质晶体结构。但是，SFX 需要在短时间内采集大量衍射图样，对高帧频探测器以及高通量数据获取提出了极高需求[3-4]。2018 年，欧洲自由电子激光装置（Eu-XFEL）通过实验首次证实了飞秒串行晶体学在 MHz 级 X 光脉冲重复频率条件下可以提供高质量的结构解析结果[5]，也说明了科学实验对 XFEL 的重复频率等性能的需求在不断提升。

XFEL 实验中获取 MHz 量级的全帧衍射图样具备极高挑战性，需要超快面探测器对图像信息进行采集，表一列出了部分 XFEL 上使用的面探测器。考虑到当前 XFEL 上的面探测器还不成熟，且没有商业产品，为应对科学实验对面探测器的高帧频、大动态范围等性能的需求，SHINE 装置规划自主研发一款电荷积分读出的、超高帧频面探测器，并命名为 STARLIGHT（SemiconducTor Array detectoR with Large dynamIc ranGe and cHarge inTegrating readout）。STARLIGHT 的目标是实现≥10kHz 的帧刷新频率，并在 12 keV 能量点实现从单光子分辨到 1 万 photons/pixel/pulse 的动态范围。

表 1 国际上现有和在研的高帧频面探测器性能比较

Table 1 Performance comparison of high frame rate area detectors

探测器 detector	像素数 number of pixels/Mpixel	帧率 frame rate/kHz	数据速率 data rate/GB·s^{-1}
CSPAD	2.3	0.12	0.5
AGIPD-1M	1.0	4500(Burst)	7.0
JUNGFRAU-4M	4.0	1.1	17.0
ePixHR	4.0	5.0	40.0
JUNGFRAU-16M	16.0	2.2	70.4
STARLIGHT-1M	1.0	10.0	20.0
STARLIGHT-4M	4.0	10.0	80.0

随着 XFEL 装置脉冲重频和面探测器帧频的提升，科学实验的数据产生速率也在不断增大，近似呈指数级增长，每两年翻一番，未来预计可达到数百 GB·s^{-1} [6]。目前，AGIPD 探测器的数据处理软件使用 FPGA 进行简单预处理和数据转发，进一步处理主要采用服务器[7]。JUNGFRAU 探测器采用了单服务器数据接收与处理方案，服务器内通过 FPGA / GPU 进行加速，依次实现了 4M 像素 1.1kHz 帧频下 9 GB·s^{-1}[8]和 4M 像素 2.2kHz 帧频下 17 GB·s^{-1}[6]的数据接收及处理速率。然而，与数据率迅猛增长的速度相比，计算机硬件性能的提升幅度显然滞后，这使得基于传统的单服务器进行数据获取的方案变得越来越困难[9]。ePixHR 探测器的数据获取通过 LCLS-II 的数据系统[10]实现。为应对该探测器的超高数据通量，LCLS-II 数据系统在探测器前端部署 Data Reduction Pipeline (DRP) 系统，通过使用 FPGA、GPU 和 CPU 混合架构以及多个节点的分布式架构对数据进行压缩、图像筛选等处理，以降低整体数据通量[11]。可见，为应对面探测器不断增长的数据通量，结合节点内的 FPGA / GPU 异构加速，有必要进一步采用多节点分布式方案进行横向扩展。

本论文将围绕面探测器数据获取与分析的挑战，结合 SHINE 自研探测器 STARLIGHT 的实际需求，提出一个分布式软件架构，以实现对该面探测器超高通量数据的高速接收和实时处理。目前本研究先只采用 CPU 处理器进行数据采集。本文余下章节分为以下几个部分，第 1 节介绍面探测器数据获取软件，第 2 节介绍数据获取软件架构与开发，第 3 节给了数据获取软件的测试结果。

1 STARLIGHT 及其数据获取需求

目前应用于XFEL的百万像素面探测器，例如运行在 LCLS的CSPAD、Eu-XFEL的AGIPD，以及SHINE 正在研发的 STARLIGHT 均采用模块化设计[12]。STARLIGHT 探测器单模块像素数量为 2×8×128×128，帧频可达到为 10kHz，预计每个像素的数据占据 2 个字节，对应每个前端模块的原始数据率约为 5 GB·s^{-1}。STARLIGHT-1M 包含 4 个模块，数据量为 20 GB·s^{-1}；而 STARLIGHT-4M 则包含 16 个模块，数据量将达到 80 GB·s^{-1}。该软件的性能指标至少要求达到 STARLIGHT-1M 的标准；为了增加其可扩展性，设计结构则按照 STARLIGHT-4M 进行设计。图一展示了 STARLIGHT 数据获取系统的硬件结构。

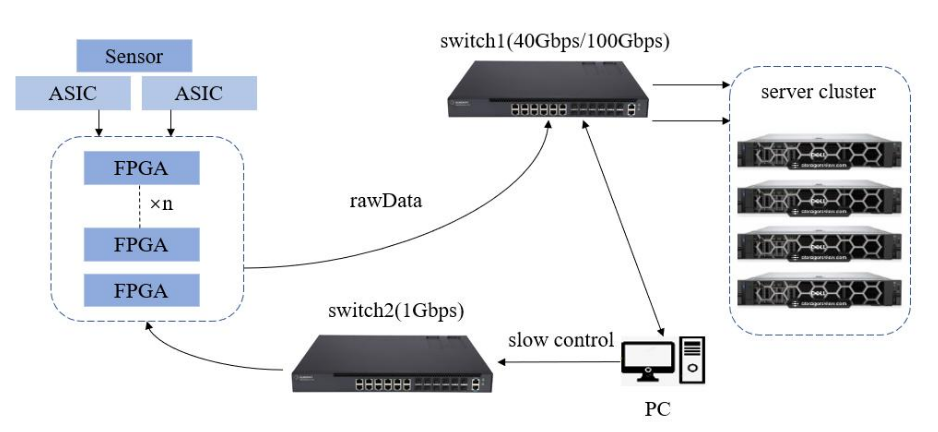


图 1 STARLIGHT 面探测器的数据获取系统的硬件架构

Fig 1 DAQ hardware architecture of STARLIGHT area detector

为实现 STARLIGHT 探测器的高通量在线数据获取，其数据获取软件需要持续高速接收来自探测器各模块的原始实验数据。数据接收节点与后端读出电子学之间的数据传输通道需要根据性能及实验要求选取标准网络传输协议，例如 UDP 和 TCP。面探测器通过多个前端模块进行数据获取，每个前端模块发送的图像数据仅包含完整实验图像的部分信息，导致每个数据接收节点只能接收不完整的数据。这导致对分布于不同节点的同一张图像数据进行事例筛选、在线分析等处理较为困难，因此需要进行在线事例重建，以将各接收模块的图像数据重新分发并汇集、组合成一张完整图像。为达成高通量实时存储的目标，软件需要具备对数据进行实时压缩处理的能力。此外，为提高探测器的用户友好度，软件还需要为用户提供实时显示运行信息及调节运行参数的图形用户界面（GUI）。

2 软件整体架构设计与模块划分

软件系统分为数据获取与处理以及前端显示两部分，其中数据获取与处理软件采用分布式结构，在结构上借鉴了 Map-Reduce 和 spark 的理念，由多个独立运行的服务器分担数据的传输和处理，充分利用多服务器的计算与内存资源、采用流水线式并行，可大幅提升数据处理的整体通量[13-14]。软件的各模块相互独立，通过监控模块实现状态同步和实时控制，各模块间利用 WebSocket 进行命令传递和监控信息的交互。前端显示软件提供控制和显示界面，可抽取部分图像数据进行可视化显示。整体架构如图二所示。

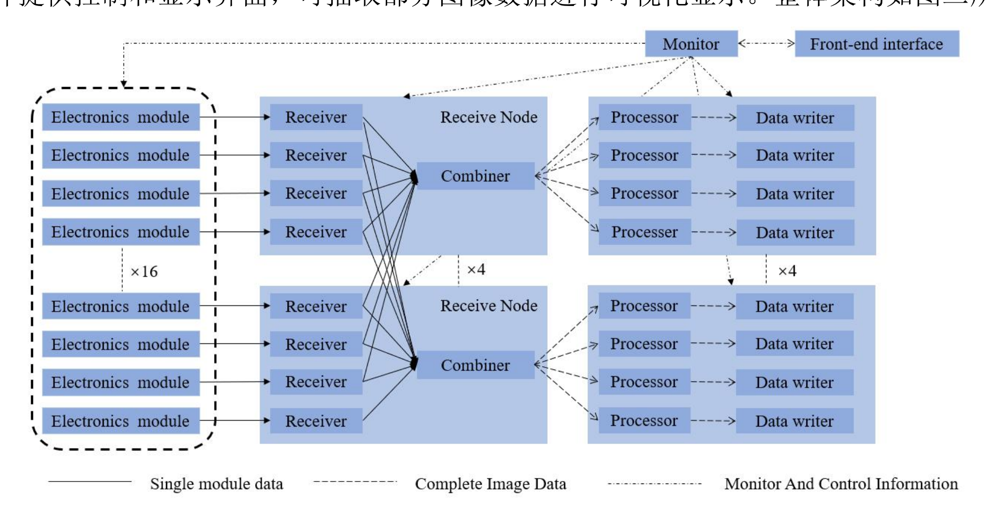


图 2 整体软件架构和数据流

Fig 2 Overall software architecture and data flow

在数据获取与处理的阶段，多个数据接收模块（Receiver）直接接收读出板数据，并将其分发至数据重建模块（Combiner），用于事例重建。事例重建的目标是将同一时刻不同探测器模块采集到的部分图像组合成完整图像。数据进一步传递至数据处理模块，进行在线刻度、压缩和存储写入等处理。

软件内部节点间采用 ZeroMQ（ZMQ）作为高通量数据流中间传输协议。该协议具有低延迟、高吞吐量和“零拷贝”等传输优势，适合高速传输大量数据。

数据获取软件的设计主要分为三个模块：数据接收模块（Receiver）、事例重建模块（Combiner）和数据预处理模块（Processor），其中包括数据刻度和实时压缩。

2.1 数据接收模块

探测器 DAQ 软件的数据接收模块是至关重要的组成部分，通过以太网交换机与各探测器模块的后端数据读出电子学直接连接。其主要职责包括数据接收、缓存和数据转发。数据接收模块可同时部署在多个数据接收节点上，以在高数据通量的环境中并行地接收数据。

数据接收节点与后端数据读出电子学之间的数据传输协议可以选择标准网络传输协议，如 UDP 或 TCP。UDP 是一种非连接协议，无需在数据传输前建立连接，具有较小的系统资源要求。尽管 UDP 可能导致数据丢失，但其实现相对简单。与之相比，TCP 是一种面向连接的协议，需要建立可靠的连接。

根据探测器硬件配置，可以在 UDP 和 TCP 之间进行选择。一些探测器（如 JUNGFRAU）的前端电子学目前采用 UDP 协议。在该协议下可能会丢失数据，对于丢失的数据包，软件上可通过将丢失位置补充为空白数据的方式进行处理[15]。对图像完整性要求高的实验，如 X 射线光子关联谱（XPCS）实验，数据丢失会产生显著影响。对于正在研发的 STARLIGHT 探测器，鉴于实验对图像完整性的高需求，选择 TCP 进行传输更为合适。

2.2 事例重建模块

图像数据的事例重建可以依据两种关键信息，一是探测器各模块内部的统一时间戳信息；二是由装置定时系统提供的 X 射线脉冲编号信息（Bunch ID）。事例重建即是将具有相同时间戳或 Bunch ID 的部分图像数据合并到一起。

在 Eu-XFEL 中，事例重建在数据传入计算集群前完成，采用了名为 Train Builder 的电子学硬件，通过多个 FPGA 板卡对图像数据进行重组[16]。参考 Map-Reduce 中的数据 Shuffle 过程，本文将给出一种基于服务器和交换机的事例重建软件实现方案[17]。它参考了 Map 和 Reduce 中数据跨节点交换的过程，可充分利用每个节点的网络带宽。

如图三所示，事例重建模块接收来自所有数据接收模块分发的数据，依据原始数据包中的 Bunch ID 或时间戳，将同一时间采集的不同模块的图像数据合并成一张完整图像（事例）。随后，将完整图像发送给数据处理模块进行后续处理。在运行时，为了实现负载均衡，数据接收模块在分发数据时可以先对 Bunch ID 进行哈希运算，然后对事例重建模块的数量取余，根据余数将数据发送到对应的事例重建模块中，以确保所有数据尽可能均匀地分发到不同的节点，如公式 1 所示。

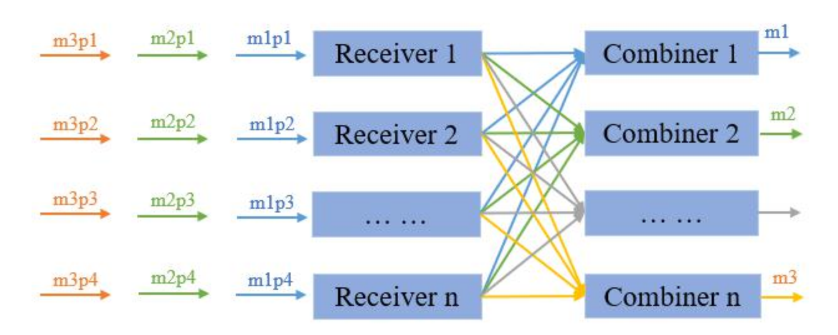


图 3 事例重建架构设计

Fig 3 Design of event reconstruction

$$i = \text{hash}\,(\text{BunchID}) \bmod n \tag{1}$$

事例重建模块采用多线程方式，每个线程负责从相应数据接收模块接收数据，将其放入与各探测器模块对应的部分图像缓存队列中。然后，将具有相同 Bunch ID 的各模块图像进行合并，形成完整的图像数据。进一步，将完整图像放入另一队列中，等待数据处理模块做后续处理。为满足超高通量数据处理的需求，事例重建软件通常以分布式的方式部署在多台服务器协同运行。

2.3 数据预处理模块

为了压低整体数据通量以尽可能降低存储的压力，需要进行数据的在线预处理，包括数据刻度、数据压缩和筛选、数据的格式化存储等。本文对其中的在线数据刻度和压缩算法进行了研究和测试。

2.3.1 数据刻度

从探测器模块直接传输过来的每个像素的信号幅度数据为原始 ADC 数据，它包含了各电子学通道的基线（pedestal）和增益（gain）等不均匀性。所以在数据处理时需要扣除基线部分，并对增益的不均匀性等进行修正，以还原出沉积能量或光子数物理量[18]。参考 JUNGFRAU、AGIPD 等面探测器，该修正可由公式（2）表示[19]，其中，ADU 表示模数转换器（ADC）单位，即模拟信号被转换为数字信号的幅度。

$$\text{energy}\,[keV] = \frac{\text{ADC}\,[ADU] - \text{pedestal}\,[ADU]}{\text{gain}\,[ADU/keV]} \tag{2}$$

基于公式（2），数据刻度模块需对每张图像的所有像素的 ADC 数据进行转换。首先，从文件读取各通道的刻度参数数据，包括各级增益和对应的基线参数，并将其存入相应的结构体中。随后，按照公式（2）对原始数据中的各通道 ADC 值进行计算，以还原出各像素的沉积能量。需要说明的是，探测器的基线、

增益等刻度参数本身的测量需要进行专门的研究和实验，本文暂不涉及。为测试软件性能，本文目前先采用模拟刻度参数进行测试。

2.3.2 数据压缩

数据压缩方面，JUNGFRAU 采用了 bitshuffle 数据预处理方法，如图四所示，先对数据进行重新排列，再结合后续的压缩算法完成压缩。在实验过程中，一般情况下探测器的大部区域信号较弱，特别是距离中心位置较远的区域。因此面探测器获取的衍射图像大部分区域的像素幅度值相对均匀且数值较小，即每个 16 位二进制数的高位基本为 0。根据该数据特点，bitshuffle 通过对图像数据进行位级别的分块转置，可以将大量位于高位的 0 位合并到完整字节中。该过程不会改变原始数据的大小，但能大大提高数据内容的重复度。结合依赖于索引的字典压缩，此方法可以大幅提高压缩效率。JUNGFRAU 的压缩流程分为两步：使用 FPGA 模块完成 bitshuffle 转置操作，然后在 CPU 上由软件进一步使用通用的压缩算法完成压缩操作[8]。

```
0000 0101                          0000 0000
0000 1001                          0000 0000
0000 0001                          0000 0000
0000 0100                          0000 1000
0001 1111   ---bit level transpose-->   0100 1101
0000 1000                          1001 1101
0000 0010                          0000 1010
0000 1001                          1110 1001
```

图 4 bitshuffle 流程

Fig 4 bitshuffle process

本文在软件层面对该压缩算法进行了测试。bitshuffle 算法的实现采用了开源软件库[20]，进一步的压缩采用了 LZ4 和 ZSTD 两种常用算法库[21-22]。

3 软件性能测试与调优

鉴于 STARLIGHT 探测器尚处于研发阶段，目前还未能产生原始数据，本文采用模拟发送实验数据的方式进行数据获取软件的性能测试。模拟数据选用公开发表的 CXIDB ID 83 数据集，该数据集来自欧洲自由电子激光装置基于 AGPID 探测器开展的飞秒串行晶体学实验[23]。

表二显示了分布式测试和单节点性能测试分别采用的服务器集群的配置参数。分布式测试集群包含 8 台服务器（以下简称集群 A），其中 4 台用于模拟数据发送，另外 4 台用于数据接收与处理；为更好地测试和评估软件性能，服务器间的数据交换物理层采用了 100G InfiniBand (IB) 网络，软件上通过 IPoIB 模块实现类以太网数据传输[24]。单节点性能测试集群包含 5 台服务器（一下简称集群 B），其中 4 台具备较大的内存，用作数据发送服务器，1 台用作数据接收和处理服务器；集群 B 的服务器采用一台 25Gbps 以太网交换机互联，4 台数据发送服务器各具有 25Gbps 的带宽，数据接收服务器具备 100Gbps 带宽。

表 2 测试服务器配置

Table 2 Test server configuration

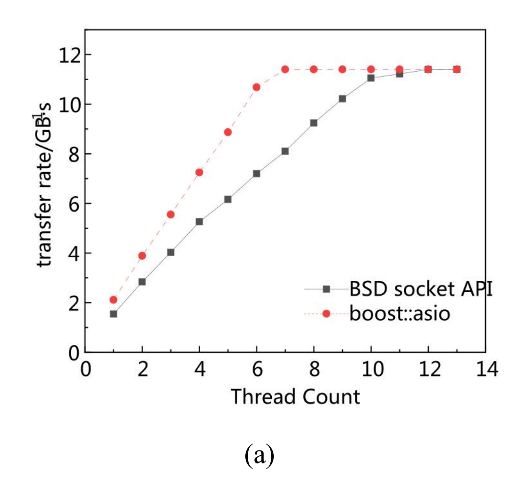
服务器集群 Server cluster	服务器用途 Server role	CPU 型号 CPU model	内存大小 memory size/GB	网卡带宽 network card bandwidth/Gbps
集群 A Cluster A	分布式测试 Distributed testing	Intel Xeon 6132	192	100
集群 B Cluster B	性能测试（接收） Performance testing(receive)	Intel Xeon 6342	251	100
	性能测试（发送）	Intel Xeon 6334	503	25

	Performance testing(sending)	

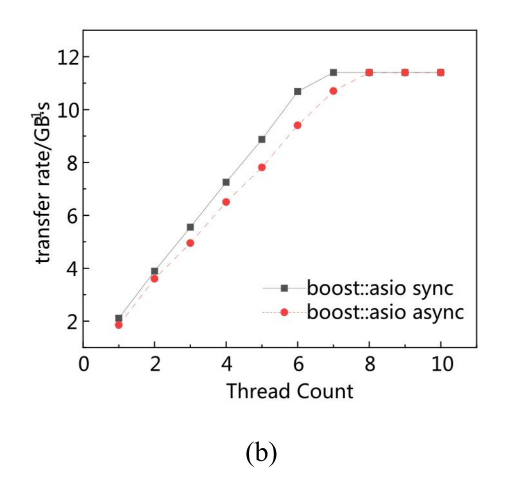
3.1 数据传输

在数据传输方面，本研究在主要集群 A 上对多种数据传输方式进行了性能测试和比较。本研究主要测试了 TCP 数据传输，分别包括采用 BSD socket API、boost::asio 库以及线程的固定 CPU 核绑定这几种方式，并对基于 boost::asio 库的同步和异步模式进行了比较。

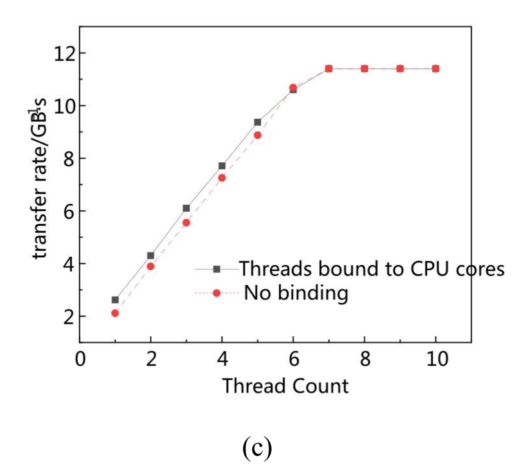
如图五(a)所示，直接调用 BSD socket API 的测试性能随着线程数量的增加逐渐上升，但总体速率较慢，4 个线程的同步接收速率大约在 5.2GB·s^{-1}。当采用 boost::asio 库后性能有所改善，特别是在多线程情况下改善明显，4 个线程时即可达到 7.3GB·s^{-1} 的接收速率。如图五(b)所示，在使用 boost::asio 库在同步和异步模式数据传输速率对比中，可以看到同步模式传输的速率优于异步模式。如图五(c)所示，线程绑定到固定 CPU 核，通过减少线程间的竞争和上下文切换来提高数据接收效率，进一步提升了性能，将四个线程的同步数据接收速率提升至 8GB·s^{-1}。



(a)



(b)



(c)

图 5 TCP 数据传输性能测试 (a)采用 BSD socket API 与 boost::asio 库在同步模式时的速率对比(b) 采用 boost::asio 库时同步和异步模式下速率对比 (c) 采用 boost::asio 库时同步模式传输有无线程 CPU 核绑定的对比

Fig 5 TCP-based performance test (a) Comparison of the speed of synchronous transmission between kernel API direct call and boost library (b) Comparison of synchronous and asynchronous speed under boost library (c) Comparison of synchronous transmission with and without thread binding under boost library

性能测试结果表明，使用 boost::asio 库相较于直接调 BSD socket API，传输速率更优。同时，在采用 boost::asio 库时，同步模式下的数据传输通量更高，更符合对探测器各模块原始数据的实时高速接收。另外，将线程绑定到固定 CPU 核心可降低线程切换开销，可进一步提高数据接收性能。

本研究也采用数据接收节点性能更好的集群 B 进行了测试。在绑定单个 CPU 核心、使用 boost::asio 库的情况下，同步 TCP 数据传输时的单线程接收速率可达 2.98GB·s^{-1}，几乎达到了数据发送网卡的带宽上限（25Gbps）。进一步使用四台服务器模拟探测器模块进行多线程数据发送和接收测试，在 4 个线程下，数据传输速率即可达到 11.6GB·s^{-1}，接近数据接收网卡的带宽上限（100Gbps）。

如上所述，STARLIGHT 探测器单模块的原始数据率可达到 5GB·s^{-1}。根据测试结果，该数据接收速率单线程很难实现，但可以通过 2 个线程同时接收实现。这需要电子学数据输出端支持单口多连接。

3.2 事例重建

本研究进一步对软件的数据接收及事例重建模块的运行进行了联合测试。下面分单节点测试和多节点测试两种情况进行介绍。

3.2.1 单节点测试

采用 CXIDB ID 83 数据集生成的 AGIPD 探测器模拟数据，在集群 B 的四台数据发送服务器中共运行 16 个模拟数据发送软件，模拟 16 个探测器模块向该集群的接收服务器持续发送原始图像数据。在数据接收服务器中同时运行数据接收模块和事例重建模块，实时进行数据接收和事例重建，软件持续运行 30 分钟。测试过程中，数据传输的总速率维持在 11.51GB·s^{-1}，达到网络带宽上限，CPU 仍有 30%以上的盈余。这说明在线事例重建的速率不低于数据接收的速率，且 CPU 资源利用较低。

3.2.2 多节点测试

为测试在多节点分布式环境下软件的工作性能和稳定性，进一步采用了集群 A 对该软件的数据接收、事例重建模块进行了测试。利用该集群进行测试时，使用其中四个节点模拟探测器数据发送，另外四个节点进行数据接收和事例重建，实测四个节点并行数据接收及事例重建总速率约为 23.5GB·s^{-1}。通过网卡带宽利用的监测发现，数据接收节点的带宽大约为数据发送节点带宽的 1.75 倍。这是由于事例重建节点和数据接收节点部署在相同的服务器上，数据接收和事例重建共同消耗同一网卡的带宽，总体占据的带宽会高于单独接收的带宽。图六显示了上述测试中单节点网卡的带宽占用情况。在事例重建过程中，在相同节点内部的数据交换无需经过网卡，而每个节点需要从网卡接收其它三个节点分发的数据。叠加来自数据发送

节点的数据流量，可以算出数据接收节点与数据发送节点的带宽占用比为 (3 + 4) / 4，即得到上述 1.75 的因子。根据如上分析，单个节点的总数据接收通量达到了约 10.3GB·s^{-1}，接近达到单个节点的网络带宽上限，这可以解释为何 4 节点的事例重建速率低于预期的 40GB·s^{-1} 以上速率。实际部署时，数据接收会采用以太网，事例重建可采用专用于集群内部数据交换的 IB 网，两者分离，从而可以避免该带宽限制。

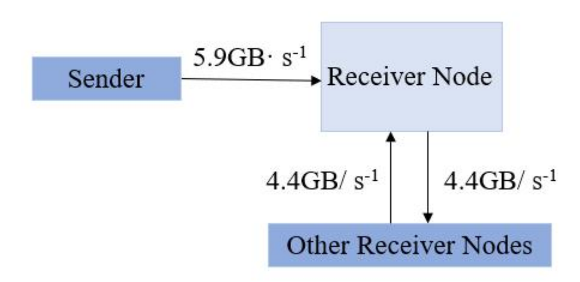


图 6 多节点事例重建测试带宽分析

Fig 6 Multi-node test bandwidth analysis

3.3 数据压缩

在数据压缩测试中，针对相同模拟数据采用 Lz4 和 ZSTD 两种不同压缩方法进行了压缩速率与压缩率的测试。ZSTD 压缩算法可以调节压缩级别，级别越高，压缩速度越慢，而压缩率则越高。本研究在进行 bitshuffle 结合 ZSTD 压缩测试时，压缩级别分别采用了 1、5、10 三种。测试结果见表三，其中括号内的数字为 ZSTD 的压缩级别。如表三所示，在使用 bitshuffle 和 Lz4 进行压缩时，相较于 ZSTD 具有更快的压缩速率，但压缩率较低；而使用 bitshuffle 和 ZSTD 时，则具有更高的压缩率，但压缩速率降低明显。另外，在单独使用 Lz4 进行压缩时，压缩后的文件较源文件更大，可能原因一方面是 Lz4 没有对原始数据进行有效压缩，同时会增加一些字典索引等额外数据。

表 3 压缩速率与压缩率测试

Table 3 Test for Compression rate and compression ratio

压缩方式 compress method	压缩速度 compress speed//MB·s^{-1}	压缩率 compress ratio
bitshuffle + Lz4	771.2	1.98
bitshuffle + ZSTD (1)	432.7	3.91
bitshuffle + ZSTD (5)	192.7	4.49
bitshuffle + ZSTD (10)	84.6	4.78
Lz4	624.2	<1
ZSTD (5)	195.1	1.71

2023 年 JUNGFRAU 探测器相关数据处理论文中，使用 FPGA 进行 bitshuffle 处理，结合服务器上的 Lz4 算法压缩，实现了 5.7 的压缩率[5]。在本文的测试中发现，实际的压缩率未能达到该文献的最高压缩率。可能的原因是，在实际实验中压缩率会受到原始图像数据本身的特性影响。例如图像整体信号幅度的大小、探测器本身各像素的噪声大小等，均会影响 bitshuffle 后的数据重复度，进一步影响压缩率。

3.4 前端界面

前端界面模块可抽取和实时显示部分图像，收集运行时信息并监视软件的运行状态。此外，该模块负责向其它软件模块发送运行时参数，包括配置文件等，以进行在线参数调节。例如压缩算法和压缩级别选择、数据文件的存储格式和路径等。与此同时，该模块还可以与后端电子学进行通信，以发送指令进行探测器的慢控。本研究对该界面进行了前期测试开发，图七给出了一个示例。图七中显示了经过事例重建和刻度后的一张完整的 AGIPD 探测器的探测图像数据。测试数据来自 CXIDB ID 83 数据集。

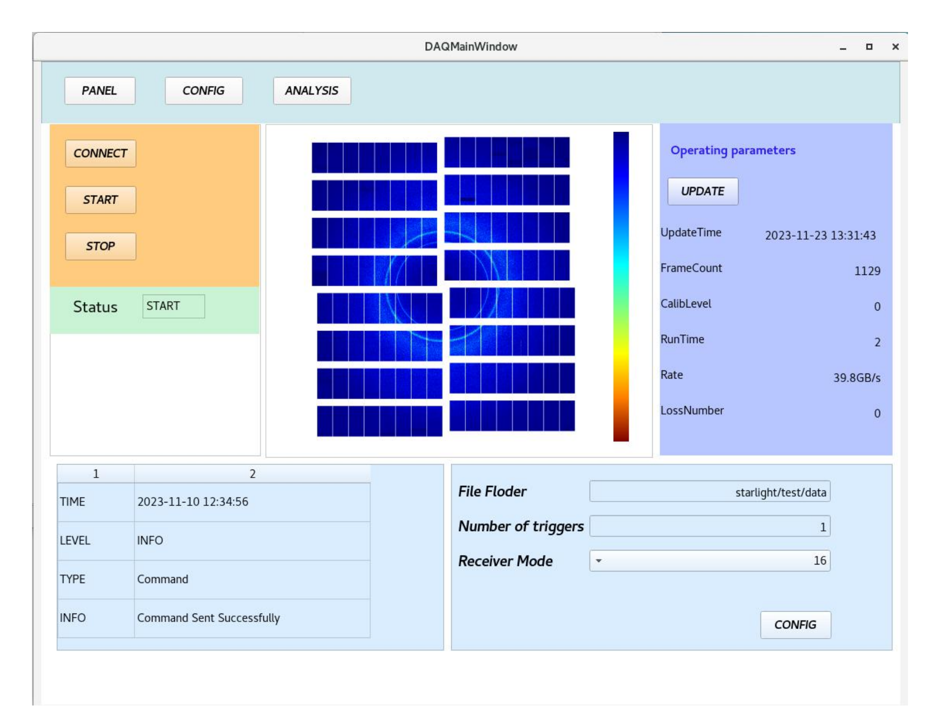


图 7 前端界面示例

Fig 7 Front-end interface test

3.5 软件整体运行测试

软件整体运行测试包括上述的数据传输、事例重建、数据刻度和压缩的全流程测试。本研究分别对其进行了单节点性能测试与多节点分布式测试，所采用的硬件配置与事例重建的测试相同。在多节点分布式测试中，集群 A 中的四台发送服务器每台各部署四个数据发送软件进程，持续以 5000 帧每秒的设定速率读取数据包并发送出去，四台数据接收服务器各部署四个数据接收软件进程、一个事例重建软件进程以及四个数据预处理软件进程。软件运行过程中的 CPU 平均使用率约为 70%，整体的数据处理速率约为 8.6GB·s^{-1}；在单节点性能测试中，集群 B 的一台数据接收服务器在事例重建测试的基础上同时部署了四个数据预处理软件。软件运行时 CPU 平均使用率约为 85%，整体的数据处理的速率为 3.4 GB·s^{-1}。可见，仅使用当前的服务器还很难满足面探测器达到几十 GB·s^{-1} 通量的数据处理需求。经分步测试分析，发现数据刻度在占满单个 CPU 核的情况下处理速率只能达到 200MB • s^{-1}，在总体运行中占据了过多的 CPU 资源。可以看出，目前仅使用 CPU 进行刻度与压缩很难满足需求。虽然可以通过增加服务器和 CPU 的数量提升整体数据处理通量，但面对几十甚至上百 GB·s^{-1} 的原始数据流，需要采用的服务器集群规模将较为庞大。为控制整体服务器集群规模，提高计算密度，采用 FPGA、GPU 等加速卡进行数据处理加速将是必要的。

4 总结与展望

面对高帧频面探测器超高通量数据获取和分析的挑战，为满足 SHINE 装置对高传输速率和高数据预处理性能的需求，本文基于 C++语言开展了一系列分布式的高性能数据获取和预处理软件的开发和测试研究。本文首先对数据传输的方式和软件进行了测试和对比分析，利用 boost::asio 库在同步模式下实现了单线程 2.98 GB·s^{-1} 的 TCP 数据接收速率。本文提出了一种基于交换机和服务器软件的在线事例重建方法，实现了单节点约 11.51 GB·s^{-1} 原始数据接收和在线事例重建速率，以及 4 节点约 23.5 GB·s^{-1} 的事例重建速率。测试结果表明，本文提出的基于软件的事例重建方法可以在多个节点上高速稳定运行，具有较好的横向扩展性。同时，本文对数据刻度和无损压缩等数据预处理功能进行了开发和测试，评估了 bitshuffle+Lz4/ZSTD 数据压缩方法的性能，为未来对该探测器数据获取软件的进一步开发及应用打下基础。

除此以外，软件仍存在一些值得优化和改进的部分。在数据刻度与压缩方面，目前仅采用了 CPU 计算硬件，发现这两个步骤会导致大量的 CPU 资源占用，影响整体软件的运行。为了缓解服务器的负担，未来利用加速卡，包括 FPGA/GPU 对数据刻度和压缩等计算密集型的数据处理进行加速将十分必要。

作者贡献声明

商琨琳负责软件开发，系统测试和调试、论文撰写和修改；怀平、李正恒负责超高帧频大动态范围 X 射线探测数据采集系统的软件系统设计、技术研发，论文研究方案的提出与修改指导；周悦负责技术支持、参与问题讨论；鞠旭东负责超高帧频大动态范围 X 射线探测系统的探测器方案设计、论文修改指导；刘志作为超高帧频大动态范围 X 射线探测系统项目的负责人提供研究经费支持和总体方案确定。

参考文献

1 赵振堂,冯超.X 射线自由电子激光[J].物理,2018,47(08):481-490.Zhao Zhentang, Feng Chao. X-ray free electron laser [J]. Physics, 2018, 47(08): 481-490. DOI: 10.7693

2 尹亮,曾孟麒,尹聪聪等.SHINE 束线站定时系统束团编号的数据采集[J].核技术,2023,46(06):060101. DOI：10.11889/j.0253-3219.2023.hjs.46.060101.

3 Chapman, H., Fromme, P., Barty, A. et al. Femtosecond X-ray protein nanocrystallography. Nature 470, 73–77 (2011). DOI: 10.1038/nature09750

4 千跃奇,刘波.串行晶体学数据筛选算法研究[J].核技术,2019,42(08):6-10. DOI： 10.11889/j.0253-3219.2019.hjs.42.080102.

5 Max O. Wiedorn, Dominik Oberthür, Richard Bean et al. Megahertz serial crystallography. Nature Communications, 9(1):4025, Oct 2018. DOI: 10.1038/s41467-018-06156-7

6 Filip Leonarski, Martin Brückner, Carlos Lopez-Cuenca et al. Jungfraujoch: hardware-accelerated data-acquisition system for kilohertz pixel-array X-ray detectors. J. Synchrotron Rad. (2023). 30, 227-234 DOI: 10.1107/S1600577522010268

7 Aschkan Allahgholi, J. Becker, A. Delfs, et al. The Adaptive Gain Integrating Pixel Detector at the European XFEL . Synchrotron Rad. (2019). 26, 74–82 DOI: 10.1107/S1600577518016077

8 Filip Leonarski, Aldo Mozzanica, Martin Brückner, et al. Jungfrau detector for brighter x-ray sources: Solutions for it and data science challenges in macromolecular crystallography. Struct Dyn. 2020 Feb 26;7(1):014305. DOI: 10.1063/1.5143480.

9 John L. Hennessy and David A. Patterson. 2019. A new golden age for computer architecture. Commun. ACM 62, 2 (February 2019), 48–60. 7 DOI: 10.1145/3282307

10 J. B. Thayer, Gabriella Carini, Wilko Kroeger, et al., "Building a Data System for LCLS-II," 2017 IEEE Nuclear Science Symposium and Medical Imaging Conference (NSS/MIC), Atlanta, GA, USA, 2017, pp. 1-4, DOI: 10.1109/NSSMIC.2017.8533033.

11 Ric Claus, Dan Damiani, Mikhail Dubrovin. et al. Data Reduction Activities at LCLS[R/OL]. (26-1- 2024) [16-3-2024] https://www.xfel.eu/sites/sites_custom/site_xfel/content/e51499/e60513/e63558/e215369/e218977/e218989/e236047/xfel_file236061/2024-01-26DataReductionatLCLS_eng.pdf

12 Philip Hart, Sébastien Boutet, Gabriella Carini, et al. The CSPAD megapixel x-ray camera at LCLS. X-Ray Free-Electron Lasers: Beam Diagnostics, Beamline Instrumentation, and Applications,volume 8504, pages 51 – 61., 2012. DOI: 10.1117/12.930924

13 M. Zaharia, M. Chowdhury, T. Das, et al. "Resilient distributed datasets: a fault-tolerant abstraction for in-memory cluster computing," in Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation, pp. 15-28, 2012. DOI:10.5555/2228298.2228301.

14 卢晓旭,顾旻皓,朱科军等.LHAASO 地面簇射粒子阵列在线实时分布式数据处理[J].核技术,2020,43(04):76-84. DOI：10.11889/j.0253-3219.2020.hjs.43.040402.

15 Dhanya Thattil, Erik Fröjdh. Sls Detector Package: source code [CP/OL]. [12-3-2022] https://github.com/slsdetectorgroup/slsDetectorPackage.

16 J Coughlan, C Day, S Galagedera, and R Halsall. The train builder data acquisition system for the european-XFEL. Journal of Instrumentation, 6(11):C11029–C11029, nov 2011. DOI: 10.1088/1748-0221/6/11/C11029

17 Jeffrey Dean and Sanjay Ghemawat. Mapreduce: Simplified data processing on large clusters.In OSDI'04: Sixth Symposium on Operating System Design and Implementation, pages 137–150, San Francisco, CA, 2004 DOI: 10.1145/1327452.1327492

18 S. Redford, M. Andrä, R. Barten, et al. Operation and performance of the JUNGFRAU photon detector during first FEL and synchrotron experiments. Journal of Instrumentation,13(11):C11006–C11006, nov 2018.DOI: 10.1088/1748-0221/13/11/C11006

19 A. Allahgholi, J. Becker, L. Bianco, et al. AGIPD, a high dynamic range fast detector for the european XFEL. Journal of Instrumentation, C01023 jan 2015 DOI: 10.1088/1748-0221/10/01/C01023

20 Kiyoshi Masui, Richard Shaw, Takeshi Yamamuro. et al. Bitshuffle : source code [CP/OL]. [27-12-2022] https://github.com/kiyo-masui/bitshuffle.

21 Yann Collet, Przemyslaw Skibinski, Takayuki Matsuoka. et al. LZ4: source code [CP/OL]. [20-1-2023] https://lz4.github.io/lz4.

22 David Reiss, Will Bailey, Pieter De Baets. et al. Zstandard: source code [CP/OL]. [22-1-2023]https://facebook.github.io/zstd.

23 Anton Barty. CXIDB ID 83: Experimental data set [DS/OL]. [11-10-2022] https://cxidb.org/id-83.html.

24 R. E. Grant, M. J. Rashti and A. Afsahi, "An Analysis of QoS Provisioning for Sockets Direct Protocol vs. IPoIB over Modern InfiniBand Networks," 2008 International Conference on Parallel Processing - Workshops, Portland, OR, USA, 2008, 79-86, DOI: 10.1109/ICPP-W.2008.25.